BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

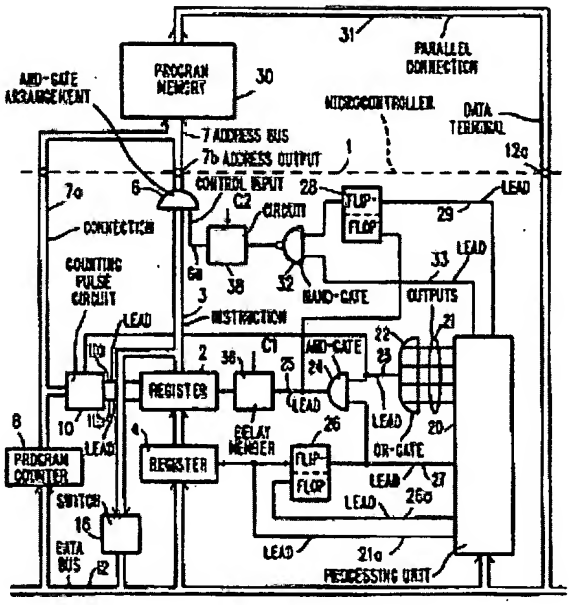
Program memory expansion using a special-function register

Patent number: DE4340551
Publication date: 1995-06-01
Inventor: GAUBATZ GERALD (DE)
Applicant: PHILIPS PATENTVERWALTUNG (DE)
Classification:
- international: G06F12/06
- european: G06F9/32, G06F9/32A
Application number: DE19934340551 19931129
Priority number(s): DE19934340551 19931129

Also published as:
EP0655688 (A2)
US5734857 (A1)
JP7262162 (A)
EP0655688 (A3)
EP0655688 (B1)

Abstract not available for DE4340551
Abstract of correspondent: **US5734857**

In order to expand the address range of the program counter in a microprocessor, a register is provided as an additional special-function register which actually consists of two separate registers which are connected in series. One register supplies the most-significant address bits via an AND-gate and is also coupled to the internal data bus. The register contents are automatically changed when the program counter passes the end position or the starting position. In the case of program jumps, the old contents of the register are also stored in the stack; this requires only few additional instructions from the standard instruction set. In the case of an interrupt, the AND-gate is blocked, so that automatically the first segment of the program memory with the interrupt programs is addressed. A linear expansion of the program address range is thus achieved without degrading the compatibility.



Data supplied from the ***esp@cenet*** database - Worldwide

(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENTAMT

(12) **Offenlegungsschrift**

(10) **DE 43 40 551 A1**

(51) Int. Cl.[6]: G 06 F 12/06

(21) Aktenzeichen: P 43 40 551.7
(22) Anmeldetag: 29. 11. 93
(43) Offenlegungstag: 1. 6. 95

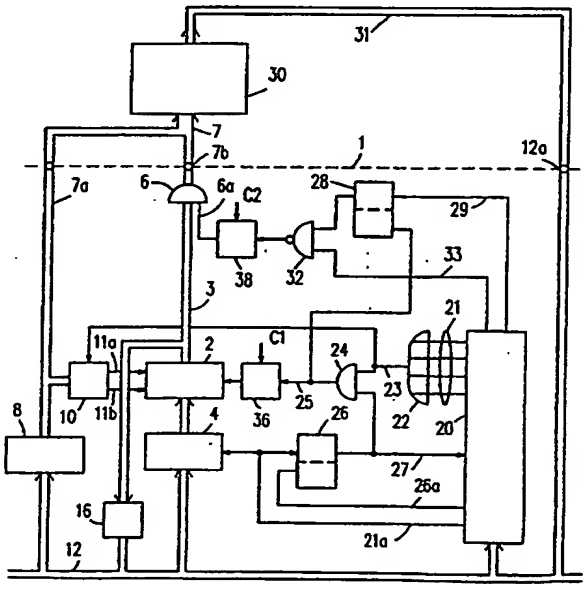
DE 43 40 551 A1

(71) Anmelder:
Philips Patentverwaltung GmbH, 20097 Hamburg, DE

(72) Erfinder:
Gaubatz, Gerald, 21149 Hamburg, DE

(54) Programmspeichererweiterung für einen Mikroprozessor

(57) Bei einem Mikroprozessor wird zur Erweiterung des Adreßbereichs des Programmzählers ein Register als zusätzliches Spezial-Funktions-Register vorgesehen, das tatsächlich aus zwei einzelnen Registern besteht, die in Reihe geschaltet sind. Das erste Register ist mit dem internen Datenbus verbunden, und das zweite Register liefert über einen zusätzlichen Ausgang (port) die höchsten Adreßbits. Bei einem linearen Überschreiten der Zählstellungsgrenzen des vorhandenen Programmzählers oder bei kleinen relativen Sprüngen wird automatisch ein Übertragungssignal bzw. Borgsignal für das zusätzliche Register erzeugt, so daß im Programm keine besonderen Befehle vorgesehen werden müssen. Lediglich für Sprünge über die ursprünglichen Adreßbereichsgrenzen müssen einige zusätzliche Befehle verwendet werden, die jedoch aus dem normalen Befehlsvorrat stammen. Auf diese Weise wird eine lineare Erweiterung des Programm-Adreßbereichs geschaffen, ohne die Kompatibilität zu verletzen.



DE 43 40 551 A1

Beschreibung

Die Erfindung betrifft einen Mikroprozessor mit einer Verarbeitungseinheit zum Empfangen und Entschlüsseln von Befehlen, einem Befehlsspeicher zur Abgabe von Befehlen und einem Programmzähler vorgegebener Kapazität zum Adressieren des Befehlsspeichers.

Derartige Mikroprozessoren sind allgemein bekannt. Eine Gruppe von Mikroprozessoren, die auch als Mikrocontroller bezeichnet werden, enthält außer dem Prozessorkern noch weitere Peripherieschaltungen, die auf demselben Chip integriert sind, und zwar insbesondere Programm- und Datenspeicher. Der Programmspeicher ist häufig als maskenprogrammierter Festwertspeicher ausgeführt, da derartige Mikrocontroller für jeweils bestimmte Anwendungen hergestellt werden, durch die das auszuführende Programm zumindest weitestgehend festgelegt ist. Derartige maskenprogrammierte Speicher benötigen nur relativ wenig Platz für jede Speicherzelle, so daß umfangreiche Befehlsspeicher mitintegriert werden können.

Für eine Gruppe von Mikrocontrollern, die von einem Basistyp mit der Bezeichnung 8051 abgeleitet sind, ist beispielsweise ein Programmspeicher mit einer Größe von 64K Speicherplätzen adressierbar. Diese maximale adressierbare Speichergröße ist vor allem dadurch gegeben, daß die Adressen 2 Byte mit je 8 Bit umfassen, was an den Prozessorkern gut angepaßt ist, da dieser jeweils ein Byte parallel verarbeitet. Außerdem sind zur Angabe von Adreßsprüngen in Sprungbefehlen nur zwei weitere Byte für die neue Adresse vorgesehen.

Durch die Integration von immer mehr Peripherieschaltungen und die Verwendung immer höherer Taktfrequenzen ergeben sich für moderne Mikrocontroller immer mehr Anwendungsmöglichkeiten, für die teilweise jedoch ein Programmspeicher mit 64K Speicherplätzen nicht mehr ausreichend ist. Eine Erweiterung des Adreßbereichs des Programmspeichers ist jedoch nicht ohne weiteres möglich, wenn vermieden werden soll, daß eine zumindest teilweise neue Befehlsstruktur erforderlich wird, da dann bereits vorhandene Programme nicht weiter verwendet werden können und auch für die Erstellung neuer Programme, die üblicherweise in einer höheren Programmsprache geschrieben und mittels Compiler in Maschinenbefehle umgesetzt werden, eine wesentliche Änderung dieser Compiler erforderlich wäre.

Aufgabe der Erfindung ist es daher, einen Mikroprozessor der eingangs genannten Art anzugeben, mit dem ein größerer Programmspeicher ansteuerbar ist, ohne die bisherige Befehlsstruktur zu verändern, so daß bereits vorhandene Programme auch weiter verwendet werden können.

Diese Aufgabe wird erfindungsgemäß dadurch gelöst, daß zur Erweiterung der maximalen Größe des Befehlsspeichers ein erstes Register, von dem parallele Ausgänge mit einem Adreßausgang des Mikroprozessors für zusätzliche höchstwertige Adreßbits des Befehlsspeichers gekoppelt sind, sowie eine Zähltakt-Schaltung vorgesehen sind und die Zähltakt-Schaltung an Adreßausgänge des Programmzählers angeschlossen ist und einen Aufwärts-Zähltakt für das erste Register erzeugt, wenn der Programmzähler seine höchste Stellung überschreitet, und einen Abwärts-Zähltakt für das erste Register erzeugt, wenn der Programmzähler seine niedrigste Stellung unterschreitet, und daß ein zweites Register vorgesehen ist, von dem parallele Eingänge an einen internen Bus angeschlossen sind zum Übernehmen des auf dem internen Bus vorhandenen Werts bei einem vorgegebenen Schreibbefehl und von dem parallele Ausgänge mit parallelen Eingängen des ersten Registers verbunden sind zur Übertragung des Inhalts des zweiten Registers in das erste Register bei Adreßsprungbefehlen.

Durch die Verwendung von zwei zusätzlichen Registern, die nach außen als ein einziges Register erscheint, das über den internen Bus geladen werden kann und für dessen Adressierung vorzugsweise eine nicht benutzte Adresse der Spezial-Funktions-Register verwendet werden kann, wird die Möglichkeit geschaffen, insgesamt 3 Byte als Adresse für den Befehlsspeicher zur Verfügung zu stellen, wobei das höchstwertige Byte gesondert verarbeitet werden kann. Wenn innerhalb eines Programms eine durch die bisher verwendeten 2 Adreßbytes gegebene Adreßgrenze überschritten wird, wird automatisch das dritte Byte erhöht bzw. erniedrigt. Letzteres kann im Rahmen von relativen Adreßsprüngen auftreten, wenn z. B. eine Programmschleife mehrmals durchlaufen wird und diese Programmschleife den durch 2 Adreßbytes gegebenen Adreßbereich überschreitet. In diesem Falle braucht bei der Programmierung auf die Erweiterung des Adreßbereichs des Befehlsspeichers keine Rücksicht genommen zu werden, da der Übergang von einem Adreßsegment zum nächsten, wenn der durch 2 Adreßbytes adressierbare Programmspeicherbereich als Segment bezeichnet wird, durch Hardware-Mittel automatisch gesteuert wird. Lediglich bei absoluten Adreßsprüngen über die Segmentgrenzen hinweg müssen zusätzlich Befehle verwendet werden, mit denen die zusätzlichen Register geladen werden, wobei diese Befehle jedoch aus dem vorhandenen Befehlsvorrat insbesondere zur Durchführung von Datenübertragungen innerhalb des Prozessors stammen.

Eine wichtige Funktion von Mikroprozessoren bzw. Mikrocontrollern ist die Verarbeitung von Unterbrechungssignalen. Solche Unterbrechungssignale werden bei bestimmten äußeren Zuständen oder auch z. B. von internen Timern erzeugt und bewirken, daß der Mikrocontroller bestimmte Befehlsfolgen in vorgegebenen Adreßbereichen des Programmspeichers ausführt. Dafür wird beim Auftreten eines Unterbrechungssignals ein Adreßsprung auf eine fest vorgegebene Adresse des Befehlsspeichers ausgelöst, die meist in einem sehr niedrigen Adreßbereich liegt. Um beim Auftreten eines Interrupt-Signals möglichst schnell einen Sprung auf die zugehörige Adresse des Programmspeichers zu ermöglichen, auch wenn gerade ein höheres Programmspeichersegment adressiert wird, ist daher eine Ausgestaltung der Erfindung dadurch gekennzeichnet, daß die parallelen Ausgänge des ersten Registers über UND-Gatter mit dem Adreßausgang verbunden sind und ein gemeinsamer Steuereingang der UND-Gatter mit einer Interrupt-Schaltung verbunden ist, die die UND-Gatter wenigstens zu Beginn der Verarbeitung eines Interrupt-Signals sperrt. Wenn das UND-Gatter bei einem Interrupt-Signal gesperrt wird, haben alle Bits des höchstwertigen Bytes der Adresse den Wert Null, wodurch automatisch das unterste Segment des Programmspeichers angesteuert wird, in dem die Adresse des Interrupt-Unterprogramms liegt. Als Vorteil ergibt sich dabei, daß der alte Inhalt der zusätzlichen Register erhalten bleibt und nach dem

Ende des Interrupt-Unterprogramms, wenn dieses das unterste Segment nicht verläßt, automatisch wieder zur Verfügung steht. Lediglich wenn innerhalb eines Interrupt-Unterprogramms ein Sprung in ein höheres Segment des Programmspeichers erfolgt, sind einige zusätzliche Befehle aus dem vorhandenen Befehlsvorrat notwendig, wie später erläutert wird.

Um solche Sprünge innerhalb eines Interrupt-Unterprogramms auf besonders einfache Weise zu ermöglichen, ist es nach einer weiteren Ausgestaltung der Erfindung vorgesehen, daß die Interrupt-Schaltung ein erstes Speicherglied enthält, das beim Auftreten eines Interrupt-Signals gesetzt und bei der Durchführung eines Adressensprungs über die Segmentgrenzen zurückgesetzt wird und dessen Ausgang mit dem Steuereingang der UND-Gatter über ein NAND-Gatter gekoppelt ist, von dem ein weiterer Eingang mit der Verarbeitungseinheit zum Empfangen eines Signals während der Ausführung einer Interrupt-Routine gekoppelt ist. Auf diese Weise wird bei einem solchen Sprung innerhalb des Interrupt-Unterprogramms die Sperre der UND-Gatter aufgehoben, so daß dann auch andere Segmente des Programmspeichers adressiert werden können.

Um allgemeine Unterprogramm-Aufrufe durchzuführen, muß der Zustand des Programmzählers und damit auch der zusätzlichen Register im sogenannten Stack zwischengespeichert werden. Dafür ist es nach einer weiteren Ausgestaltung der Erfindung zweckmäßig, daß der Ausgang des ersten Registers mit dem internen Bus koppelbar ist. Dadurch kann der Inhalt dieses Registers leicht abgefragt werden.

Die Zähltakt-Schaltung soll den Übergang von einem Segment des Programmspeichers in ein benachbartes Segment automatisch auslösen, so daß für den normalen Programmablauf der Programmspeicher als ein einziger zusammenhängender Adreßbereich erscheint. Dies soll auch für den Fall relativer Programmsprünge zutreffen, die über die Grenze des durch 2 Adreßbytes gegebenen Segments erfolgen. Um dies leicht zu erreichen, ist eine weitere Ausgestaltung der Erfindung dadurch gekennzeichnet, daß die Zähltakt-Schaltung mit Ausgängen für die beiden höchstwertigen Adreßbits des Programmzählers gekoppelt ist und eine zweite Speicherstufe, die den Signalzustand des höchstwertigen Adreßbits für eine vorgegebene Zeitdauer nach einem Zustandswechsel dieses Adreßbits speichert, sowie zwei Gatter mit je wenigstens drei Eingängen enthält, wobei die Eingänge jedes Gatters mit wenigstens einem Ausgang der Speicherstufe sowie mit den Ausgängen für die beiden höchstwertigen Adreßbits derart gekoppelt sind, daß das eine Gatter einen Aufwärts-Zähltakt abgibt, wenn beide Adreßbits ihren Zustand von hoch nach niedrig wechseln, und das andere Gatter einen Abwärts-Zähltakt abgibt, wenn beide Adreßbits ihren Zustand von niedrig nach hoch wechseln. Durch die Verwendung von nur der höchstwertigen beiden Bits der unteren beiden Adreßbyts ist es möglich, beim Überschreiten einer Segmentgrenze die Richtung zu unterscheiden, in der diese Grenze passiert wird, so daß automatisch richtig ein Aufwärts-Zähltakt oder Abwärts-Zähltakt abhängig von dieser Richtung erzeugt wird.

Es ist klar, daß der Programmspeicher nicht ein einziger Speicher sein muß, der mit dem Mikrocontroller auf einem Chip integriert ist, sondern er kann auch aus mehreren Teilen für jeweils verschiedene Adreßbereiche bestehen, wobei zumindest einige Teile auch außerhalb des Mikrocontrollers in getrennten Bausteinen vorhanden sein können.

Ausführungsbeispiele der Erfindung werden nachstehend näher anhand der Zeichnung erläutert. Es zeigen:

Fig. 1 ein Blockschaltbild einiger erfindungswesentlicher Teile eines Mikrocontrollers,

Fig. 2 eine Ausführungsform der Zählschaltung in **Fig. 1**,

Fig. 3 ein Diagramm zur Erläuterung der Erzeugung der Ausgangssignale der Zählschaltung.

In **Fig. 1** enthält ein Mikrocontroller 1 eine Anzahl Schaltungen bzw. Elemente, von denen hier nur einige dargestellt sind. Der Mikrocontroller 1 ist über einen Adreßbus 7 mit einem Programmspeicher 30 verbunden, der außerhalb des Mikrocontrollers angeordnet ist, der jedoch hier nur einen Teil des gesamten Programmspeichers darstellen möge, während ein weiterer, insbesondere kleinerer Teil für die untersten Adressen in dem Mikrocontroller 1 enthalten ist, was hier jedoch der Einfachheit halber nicht näher dargestellt ist. Der Datenausgang des Programmspeichers 30 ist über eine parallele Verbindung 31 mit einem Datenanschluß 12a des Mikrocontrollers 1 verbunden, der auf den Datenbus 12 führt. Eventuell vorhandene Schalter in Datenwegen sind hier der Übersichtlichkeit halber weggelassen, da sie für das Prinzip der Funktion nicht von Bedeutung sind.

An den internen Datenbus 12 ist eine Verarbeitungseinheit 20 angeschlossen, die insbesondere empfangene Befehle entschlüsselt und verschiedene Steuersignale über getrennte Steuerleitungen abgibt. Von diesen Steuerleitungen sind hier nur einige angegeben, die später erläutert werden.

Ferner ist ein Programmzähler 8 vorhanden, der über eine Verbindung 7a einen Teil der Adreßbits der Adresse für den Programmspeicher 30 liefert. Es wird angenommen, daß der Programmzähler 8 2 Byte breit ist und somit die Adreßbits A0 bis A15 über die Verbindung 7a abgibt.

Schließlich sind noch zwei Register 2 und 4 vorhanden, von denen das Register 2 Daten vom Register 4 übernehmen kann und übernommene Daten über eine Verbindung 3 und eine UND-Gatteranordnung 6 über einen Adreßausgang 7b des Mikrocontrollers 1 einen Teil der Adreßbits für den Adreßbus 7 liefert. Die Gatteranordnung 6 enthält für jede einzelne Leitung in der Verbindung 3 ein einzelnes UND-Gatter, die alle über einen Steuereingang 6a gemeinsam gesteuert werden. Der Ausgang des Registers 2 ist über die Verbindung 3 außerdem mit einem Schalter 16 verbunden, über den der Inhalt des Registers 2 auf den internen Datenbus 12 übertragen werden kann. Ferner kann der Inhalt des Registers 2 durch einen Zähltakt auf einer der Leitungen 11a und 11b um jeweils eine Einheit erhöht oder erniedrigt werden. Diese Zähltake auf den Leitungen 11a und 11b werden von einer Zähltakt-Schaltung 10 geliefert.

Ein paralleler Eingang des Registers 4 ist mit dem internen Datenbus 12 verbunden und übernimmt den Wert auf diesem internen Datenbus 12 mit einem Schreibsignal auf einer Leitung 21a, die von der Befehlsentschlüsselung in der Verarbeitungseinheit 20 kommt. Weitere Elemente, die in **Fig. 1** dargestellt sind, werden nachfolgend bei der Beschreibung verschiedener Funktionen näher erläutert.

Zunächst wird angenommen, daß ein einfaches Programm in Form einer linearen Befehlsfolge ausgeführt wird. Der Inhalt des Befehlsspeichers bzw. der Adreßbereich für die Adressierung dieses Inhalts wird in

Segmente eingeteilt, wobei der Adreßbereich eines Segments durch die Stellenzahl des Programmzählers 8 bestimmt wird, z. B. 16 Stellen entsprechend 2 Byte. Die einzelnen Segmente werden durch den Inhalt des Registers 2 bestimmt. Wenn nun das lineare Programm eine Segmentgrenze überschreitet, d. h. der Programmzähler 8 geht über seine höchste Stellung auf die Anfangsstellung zurück, wird von der Zähltakt-Schaltung 10, die mit einigen Ausgängen des Programmzählers 8 verbunden ist, ein Aufwärts-Zähltakt auf der Leitung 11a für das Register 2 erzeugt, so daß dessen Inhalt um eine Einheit weiter gezählt wird. Damit wird automatisch der Anfang des folgenden Segments im Programmspeicher adressiert, ohne daß hierfür innerhalb des Programms irgendwelche Maßnahmen erforderlich sind.

Programme enthalten jedoch normalerweise Sprünge, wobei relative und absolute Sprünge zu unterscheiden sind. Bei einem relativen Sprung wird zu dem Inhalt des Programmzählers 8 ein Wert hinzugezählt oder von diesem Inhalt subtrahiert, wobei dieser Wert in dem Befehl, vorzugsweise im zweiten Byte eines 2-Byte-Befehls angegeben ist. Wenn bei einem Aufwärts-Sprung des Programmzählers 8 seine maximale Stellung überschritten wird, d. h. eine Segmentgrenze wird überschritten, erzeugt die Zähltakt-Schaltung 10 ebenfalls einen Aufwärts-Zähltakt für das Register 2 auf der Leitung 11a. Entsprechendes gilt für einen Rücksprung, wenn eine Segmentgrenze gerade in Vorwärtsrichtung überschritten war und nun ein Rücksprung über die Segmentgrenze erfolgt. Dabei springt der Programmzähler 8 von einem niedrigen Wert auf einen hohen Wert, und die Zähltakt-Schaltung 10 erzeugt auf der Leitung 11b einen Abwärts-Zähltakt, der den Inhalt des Registers 2 um eine Einheit erniedrigt. Dabei wird davon ausgegangen, daß ein relativer Sprung nur über einen kleinen Teil des Adreßbereichs eines Segments erfolgen kann. In jedem Falle wird das Register 2 bei einem relativen Sprung automatisch in der richtigen Weise verändert, ohne daß innerhalb des Programms besondere Maßnahmen ergriffen werden müssen.

Dies ist jedoch bei absoluten Sprüngen anders. Diese können praktisch beliebig groß sein, und bei einem solchen Sprung wird die Zieladresse, an der das Programm fortgesetzt werden soll, im Sprungbefehl direkt angegeben. Ein solcher Sprungbefehl im vorhandenen Befehlsvorrat umfaßt 3 Byte, von denen die letzten beiden Bytes für die Angabe der Zieladresse vorgesehen sind. Solange ein Sprung nur innerhalb eines Segments erfolgt, wird der Inhalt des Registers 2 nicht verändert. Dazu reicht ein einfacher Sprungbefehl im Programm aus.

Wenn jedoch auch ein Sprung in ein anderes Segment erfolgen soll, muß auch der Inhalt des Registers 2 verändert werden. Dies erfolgt mit einem normalen Datentransportbefehl, wobei das Register 2 und das Register 4 als Einheit betrachtet und mit PACE bezeichnet werden, und dieses einheitliche Register PACE ist ein Spezial-Funktions-Register mit einer bei bekannten Mikrocontrollern nicht benutzten Adresse, so daß ein Adressensprung im Programm wie folgt aussieht:

```
1  MOV     PACE, #data
2  LJMP    addr16
```

Der Befehl 1 bedeutet, daß der Wert "data", der in diesem Befehl im zweiten Byte angegeben ist, in das Register PACE übertragen wird. Bei der in Fig. 1 dargestellten Schaltung bedeutet dies, daß auf der Leitung 21a ein Schreibsignal erzeugt wird, mit dem der Wert "data" in das Register 4 eingeschrieben wird, und gleichzeitig wird mit diesem Schreibbefehl ein Flipflop 26 gesetzt. Dadurch entsteht auf der Leitung 27 ein hohes Signal, mit dem ein UND-Gatter 24 freigegeben wird und das außerdem der Verarbeitungseinheit 20 zugeführt wird, wo dieses Signal verhindert, daß ein eventuell eintreffendes Interrupt-Signal unmittelbar verarbeitet wird. Dadurch wird gewährleistet, daß ein Adressensprung, der in diesem Falle zwei Befehle umfaßt, wirklich zusammenhängend ausgeführt wird, da sonst unter bestimmten Bedingungen Fehlfunktionen auftreten können.

Mit dem Befehl 2 wird der Programmzähler 8 auf die in diesem Befehl angegebene 2-Byte-Adresse gesetzt. Dabei erzeugt die Verarbeitungseinheit 20 an einem der Ausgänge 21, die entschlüsselten Sprungbefehlen zugeordnet sind, ein Signal, das über ein ODER-Gatter 22, die Leitung 23, das UND-Gatter 24 und die Leitung 25 auf ein Verzögerungsglied 36 gelangt, das dieses Signal erst mit einer bestimmten Taktphase C1 innerhalb des Befehlsablaufs wirksam werden läßt, nämlich genau zu dem Zeitpunkt, zu dem der Programmzähler 8 die neue Adresse am Ausgang wirksam werden läßt, und zu diesem Zeitpunkt wird der Inhalt des Registers 4 in das Register 2 übertragen, so daß alle Bytes der neuen Adresse gleichzeitig auf dem Adreßbus 7 erscheinen. Das Flipflop 26 wird über ein Signal von der Verarbeitungseinheit 20 über die Leitung 26a wieder zurückgesetzt. Dieses Signal erscheint bei jedem Befehl, und damit ist sichergestellt, daß normalerweise ein Interrupt-Signal sofort verarbeitet werden kann.

Die Zähltakt-Schaltung 10 wird bei absoluten Programmsprüngen nicht benötigt, jedoch muß verhindert werden, daß keine falschen Zähltakt-Signale auftreten. Wenn nämlich z. B. ein Sprung von einer Adresse am Anfang eines Segments zu einer Adresse am Ende eines Segments erfolgt, erscheint dies für die Zähltakt-Schaltung wie ein kurzer relativer Sprung rückwärts über die Segmentgrenze. Daher ist die Leitung 23, auf der durch das ODER-Gatter 22 bei jedem absoluten Sprung ein Signal auftritt, auch zur Zähltakt-Schaltung 10 geführt, um dort jedes eventuell erzeugte Zähltakt-Signal zu unterdrücken.

Eine besondere Art von absoluten Sprüngen tritt auf, wenn innerhalb eines Programmablaufs ein Unterprogramm auf gerufen werden soll, das an einer bestimmten entfernten Speicherstelle beginnt. In diesem Falle muß nämlich nach der Beendigung des Unterprogramms automatisch an die Stelle zurückgekehrt werden, von der aus das Unterprogramm aufgerufen wurde. Dazu muß die momentane Programmadresse beim Aufruf eines Unterprogramms zwischengespeichert werden, was auf dem sogenannten Stack erfolgt. Dies ist ein Speicherbereich in einem Datenspeicher, der von einer bestimmten Adressieranordnung adressiert wird. Das Zwischenspeichern des Inhalts des Programmzählers 8 beim Aufruf eines Unterprogramms erfolgt bei der Befehlsausführung automatisch, jedoch muß nun zusätzlich noch der Inhalt des Registers 2 zwischengespeichert werden. Dies

erfolgt durch die nachstehend angegebene Befehlsfolge:

```
        3       MOV     direct,PACE
        4       MOV     PACE,#data
        5       LCALL   SUBR
                .
                .
Subr:   6       PUSH    direct
                .
                .
                .
        7       POP     PACE
        8       RET
```

Mit dem Befehl 3 wird der Inhalt des Registers 2, also des mit PACE bezeichneten Registers, über den Schalter 16 ausgelesen und über den internen Datenbus 12 in eine Speicherstelle eines nicht dargestellten Datenspeichers übertragen, die durch die Adresse "direct" angegeben ist. Diese Adresse wird zweckmäßig nur für diese Programmsprünge reserviert und sonst nicht benutzt.

Mit dem Befehl 4 wird der im Befehl enthaltene Wert "data" in das Register 4 eingeschrieben. Dieser Wert gibt dasjenige Segment an, in dem das nun folgende Unterprogramm beginnt. Mit dem Befehl 5 wird schließlich der Befehlszähler 8 auf die Adresse gesetzt, an der das Unterprogramm in dem betreffenden Segment beginnt, und gleichzeitig wird wieder über das ODER-Gatter 22, das UND-Gatter 24 und den Block 36 ein Ladesignal zum Laden des Inhalts des Registers 4 in das Register 2 erzeugt. Damit steht nun auf dem Adreßbus 7 die vollständige Adresse für den Beginn des Unterprogramms zur Verfügung. Die folgenden Befehle des Hauptprogramms sind nicht von Bedeutung und daher hier nur durch Punkte angedeutet.

Im Unterprogramm ist der erste Befehl der Befehl 6, mit dem der Inhalt des Speicherplatzes "direct" in dem Stack abgespeichert wird, und zwar unmittelbar hinter dem Wert des Programmzählers 8, bei dem also das Hauptprogramm verlassen wurde. Damit ist die Rücksprungadresse in der richtigen Reihenfolge im Stack abgelegt.

Nun folgen die Befehle zur Ausführung des Unterprogramms, und am Ende tritt der Befehl 7 auf, mit dem das Segmentadreßbyte der Rücksprungadresse vom Stack in daß Register PACE geladen wird. Dies bedeutet, daß dieser Wert in das Register 4 eingeschrieben wird. Der Befehl 8 bedeutet den Rücksprung in das Hauptprogramm, indem der Programmzähler 8 wieder mit dem Inhalt der vorhergehenden Speicherplätze des Stack geladen wird, und gleichzeitig mit diesem Laden wird auch der Inhalt des Registers 4 in das Register 2 übertragen, und damit stehen wieder gleichzeitig alle Bytes der Adresse des Hauptprogramms zur Verfügung.

Es sei bemerkt, daß die Befehlsabfolge für alle segmentgrenzenüberschreitenden Unterprogrammaufrufe gleich ist, ebenso wie die willkürlich wählbare Adresse "direct", da diese nur bei der Ausführung des Befehlssprungs benötigt wird und nach dem Befehl 6 bereits wieder frei ist.

Bisher wurde davon ausgegangen, daß die UND-Gatteranordnung 6 über den gemeinsamen Steuereingang 6a freigegeben war. Dies war bei den bisher beschriebenen Befehlsabläufen der Fall, weil mindestens die Leitung 33 von der Verarbeitungsanordnung 20 ständig ein logisches Signal "Null" erhält und damit über das NAND-Gatter 32 ständig ein hohes Potential erzeugt wird, das über die Schaltung 38 direkt weitergeleitet wird. Im übrigen wird bei einem der vorher beschriebenen Programmsprünge in ein anderes Segment über das dann auftretende Signal auf der Leitung 25 ein Flipflop 28 in eine Stellung gesetzt, in der es ebenfalls ein hohes Potential an das ODER-Gatter 32 liefert.

Es muß nun auch beim Auftreten von Interrupt-Signalen ein korrekter Programmablauf gewährleistet sein. Dabei muß dann ein Programmsprung in das unterste Segment an eine bestimmte Speicherstelle erfolgen bzw., wenn mehrere verschiedene Interrupt-Signale mit u. U. unterschiedlicher Priorität verarbeitet werden können, ein Programmsprung in das unterste Segment an eine vom Interrupt-Signal abhängige Speicherstelle. Diese Speicherstellen sind fest vorgegeben. Die Interrupt-Signale werden der Verarbeitungsanordnung 20 über nicht dargestellte Eingänge zugeführt.

Wenn ein Interrupt-Signal auftritt, erscheint auf den Leitungen 29 und 33 ein hohes Signal. Damit wird das Flipflop 28 umgeschaltet und führt dem NAND-Gatter 32 ebenfalls ein hohes Signal zu. Damit empfängt die Schaltung 38 ein niedriges Signal vom NAND-Gatter 32, und bei einer bestimmten Phase C2 innerhalb des Interrupt-Befehls, nämlich wenn der Programmzähler 8 auf die Interrupt-Adresse gesetzt wird, wird dieses niedrige Signal vom ODER-Gatter 32 auch auf die Steuerleitung 6a durchgeschaltet, so daß die Gatteranordnung 6 gesperrt wird und am Ausgang 7b alle Bits den Wert "0" haben, d. h. es wird automatisch das unterste

Segment adressiert. Der Inhalt des Registers 2 bleibt dabei erhalten.

Wenn das Interrupt-Programm, das an der Interrupt-Adresse beginnt, vollständig im untersten Segment enthalten ist, wie es meistens der Fall sein wird, wird am Ende des Interrupt-Programms der Programmzähler 8 durch den Rücksprung-Befehl am Ende des Interrupt-Programms auf die Adresse des Hauptprogramms gesetzt, die der Stelle folgte, an der das Unterbrechungssignal aufgetreten ist. Gleichzeitig wird das Signal auf der Leitung 33 wieder niedrig, und damit erhält das NAND-Gatter 32 wieder ein niedriges Eingangssignal und erzeugt ein hohes Ausgangssignal, das über die Schaltung 38 und die Steuerleitung 6a auf die UND-Gatteranordnung 6 gelangt, so daß automatisch wieder dasselbe Segment adressiert wird wie an der Stelle des Hauptprogramms, an der das Unterbrechungssignal auf getreten war. Für diese Interrupt-Programme sind also keine besonderen Maßnahmen innerhalb des Programms erforderlich.

Es kann jedoch der Fall auftreten, daß ein Unterbrechungsprogramm ein Unterprogramm benötigt, das in einem anderen Segment des Programmspeichers enthalten ist. Das bedeutet, daß innerhalb des Interrupt-Programms ein Programmsprung ausgeführt werden muß, der auch den Inhalt des Registers 2 verändert. In diesem Falle muß jedoch der momentane Inhalt des Registers 2 gesichert und zurückgeladen werden, wenn das Interrupt-Programm beendet wird. Hinzu kommt, daß ein Interrupt-Signal zu beliebigen Zeiten auftreten kann, beispielsweise auch zwischen den Befehlen 3 und 4 des vorher beschriebenen normalen Programmsprungs. Lediglich zwischen den Befehlen 4 und 5 wird ein Interrupt-Befehl nicht unmittelbar ausgeführt, sondern die Ausführung beginnt dann erst nach dem Befehl 5. Wenn aber nach dem Befehl 3 ein Sprung in ein Interrupt-Programm erfolgt, erfolgt nach dessen Ausführung der Rücksprung zum Befehl 4, und das folgende Programm geht davon aus, daß an der Adresse "direct" im Befehl 3 der Inhalt des Registers 2, also PACE, enthalten ist. Daher muß der Inhalt der Adresse "direct" ebenfalls gesichert werden, denn bei einem Unterprogrammaufruf in einem anderen Segment innerhalb der Ausführung des Interrupt-Programms wird die Adresse "direct" ebenfalls verwendet.

Aus diesem Grund muß ein solches Interrupt-Programm wie folgt aufgebaut sein:

```
 9      PUSH      PACE
10      PUSH      direct
        .
        .
        .
11      MOV       direct,#0
12      MOV       PACE,#data
13      LCALL     SUBR
        .
        .
        .
14      POP       direct
15      POP       PACE
16      RETI
```

Mit dem Sprung in das Interrupt-Programm wird der vorhergehende Inhalt des Programmzählers 8 im Stack gespeichert. Der erste Befehl im Interrupt-Programm ist der Befehl 9, mit dem dann der noch vorhandene Inhalt von PACE auf dem Stack gesichert wird. Mit dem folgenden Befehl 10 wird daran anschließend im Stack auch der Inhalt der Adresse "direct" gesichert. Nun beginnt das eigentliche Interrupt-Programm, wie durch die Punkte angedeutet ist.

Die Befehle 9 und 10 können auch an anderen Stellen vor dem Befehl 11 im Interrupt-Programm stehen.

Wenn nun ein Programmsprung zu einem Unterprogramm in einem anderen Segment erfolgen soll, muß wieder im Prinzip die Befehlsfolge 3 bis 5 wie bei einem normalen Programmsprung erfolgen. Dabei muß jedoch sichergestellt werden, daß der Rücksprung aus diesem Unterprogramm wieder in das unterste Segment mit der Segmentadresse "0" erfolgt, von wo aus der Programmsprung aus dem Interrupt-Programm erfolgt. Dazu wird im Befehl 11 nun nicht der Inhalt von PACE, der sich ja auf das früher unterbrochene Programm bezieht und im übrigen im Befehl 9 bereits gesichert ist, sondern der Wert "0" (entsprechend Segment 0) geladen. Die Befehle 12 und 13 entsprechen dann unmittelbar den Befehlen 4 und 5 bei einem normalen Programmsprung.

Das Unterprogramm selbst ist hier nicht näher angedeutet, beginnt jedoch ebenfalls wie beim vorher beschriebenen normalen Programmsprung mit dem Befehl 6, mit dem der Inhalt der Adresse "direct", d. h. nun der Wert "0", in den Stack im Anschluß an die beim Befehl 13 gespeicherte Adresse des Befehlszählers 8 gespeichert wird. Am Ende dieses Unterprogramms folgt dann der Befehl 7, mit dem der Wert "0" aus dem Stack in PACE

geladen wird, und dann erfolgt der Rücksprung in das Interrupt-Programm im Segment "0". Hier wird klar, weshalb durch die Befehle 9 und 10 der ursprüngliche Inhalt von PACE und der Adresse "direct" im Stack zwischengespeichert werden muß.

Nach dem Rücksprung in das Interrupt-Hauptprogramm wird dieses fortgesetzt, und am Ende des Interrupt-Programms folgen die Befehle 14 und 15, mit denen der alte Zustand vor dem Interrupt-Signal in PACE und der Adresse "direct" wieder hergestellt wird. Mit dem Befehl 16 folgt schließlich der normale Rücksprung in das unterbrochene Hauptprogramm, das dann normal fortgesetzt werden kann.

Bei dieser Befehlsfolge laufen in der Schaltung nach Fig. 1 folgende Funktionen ab. Mit dem Sprung in das Interrupt-Programm wird, wie vorher beschrieben, die Gatteranordnung 6 gesperrt und das Flipflop 28 umgeschaltet. Die Befehle 9, 10 und 11 des Interrupt-Programms sind Datentransportbefehle, die sich auf die dargestellte Schaltung praktisch nicht auswirken.

Mit dem Befehl 12 wird der Wert "data" dem Register 4 zugeführt und durch ein bei diesem Befehl erzeugtes Schreibsignal auf der Leitung 21a in das Register 4 eingeschrieben, und gleichzeitig wird das Flipflop 26 umgeschaltet. Beim folgenden Befehl 13 wird über eine der Ausgangsleitungen 21, das ODER-Gatter 22, das nun über die Leitung 27 freigegebene UND-Gatter 24 und die Schaltung 36 ein Schreibsignal auf das Register 2 gegeben, wodurch der Inhalt des Registers 4 übernommen wird. Durch das auf der Leitung 25 vom UND-Gatter 24 erzeugte Signal wird aber außerdem das Flipflop 28 wieder zurückgeschaltet, da das Signal auf der Leitung 29 nur beim Sprung in das Interrupt-Programm erzeugt wurde. Damit erhält nun das NAND-Gatter 32 wieder an einem Eingang ein niedriges Signal, wodurch mit entsprechender Verzögerung in der Schaltung 38 die Steuerleitung 6a ein hohes Signal erhält und die Gatteranordnung 6 freigibt, so daß der in das Register 2 übertragene Inhalt am zusätzlichen Adressenausgang 7b für die Adressierung des Programmspeichers erscheint. Damit werden nun die Befehle des Unterprogramms adressiert, während das Flipflop 28 seinen Zustand beibehält und ständig die UND-Gatteranordnung 6 freigibt. Dies gilt auch, wenn am Ende des Unterprogramms der Rücksprung in das Interrupt-Programm erfolgt, und auch beim Rücksprung aus dem Interrupt-Programm in das unterbrochene Hauptprogramm. Von diesem Zeitpunkt an hat dann auch wieder die Leitung 33 ein niedriges Signal, wodurch das NAND-Gatter 32 ein zweites niedriges Signal empfängt, was jedoch am freigegebenen Zustand der UND-Gatteranordnung nichts ändert. Der Adreßausgang 7b wird also dauernd vom Inhalt des Registers 2 bestimmt.

Auf diese Weise sind Programmsprünge in beliebige Segmente des Programmspeichers möglich, wobei nur wenige zusätzliche Befehle aus dem üblichen Befehlsvorrat erforderlich sind.

Die meisten Elemente der in Fig. 1 dargestellten Schaltung, wie Register, Gatter und Flipflops, sind in ihrem Aufbau allgemein bekannt. Die Verarbeitungsanordnung 20 stellt den Kern eines üblichen Mikroprozessors dar und enthält insofern auch keine Besonderheiten. Lediglich für die Zähltakt-Schaltung 10 wird nachfolgend eine mögliche Ausführungsform anhand der Fig. 2 und 3 beschrieben.

In Fig. 2 enthält die Zähltakt-Schaltung 10 ein Flipflop 40, ein NOR-Gatter 42 und ein UND-Gatter 44 sowie zwei Inverter 46 und 48. A14 und A15 stellen die beiden höchsten Bits des Programmzählers 8 auf der Verbindung 7a in Fig. 1 dar. Über die Leitung 9 wird ein Taktsignal zugeführt, das jeweils vor einem Adreßwechsel des Programmzählers auftritt. Über die Leitung 49 wird ein weiteres Taktsignal zugeführt, das kurz nach einem Adreßwechsel des Programmzählers auftritt, also dann, wenn ein neuer Befehl aus dem Programmspeicher gelesen werden soll. Ein erzeugtes Zähltakt-Signal erscheint auf der Leitung 11a bzw. 11b. Über ein Signal auf der Leitung 23 wird die Erzeugung eines Zähltakt-Signals verhindert.

Zunächst wird angenommen, daß das Signal auf der Leitung 23 niedrig ist, so daß der Inverter 46 ein hohes Signal abgibt. Das Flipflop 40 ist ein D-Flipflop, dessen D-Eingang mit dem höchstwertigen Adreßbit A15 verbunden ist. Das Signal QN am Ausgang 41 des Flipflops 40 folgt invertiert jeweils bei der positiven Flanke des Taktsignals auf der Leitung 9 dem Verlauf des Signals auf dem höchstwertigen Adreßbit A15, wie aus Fig. 3a zu erkennen ist.

Wenn nun angenommen wird, daß der Programmzähler linear bis zu seiner höchsten Stellung aufwärts gezählt wird, wobei das Adreßbit A14 zunächst auf einen hohen Signalwert geht, den das Adreßbit A15 bereits hat, gehen nach der maximalen Stellung mit dem nächsten Taktsignal für den Programmzähler beide Adreßbit A14 und A15 praktisch gleichzeitig auf einen niedrigen Signalwert, während der Ausgang QN des Flipflops 40 noch einen niedrigen Signalwert hat. Wenn zunächst das Ausgangssignal des Inverters 48 nicht berücksichtigt wird, würde auf der mit OFL (Überlauf) bezeichneten Ausgangsleitung 11a ein Signal entstehen, wie in Fig. 3a gestrichelt gezeichnet ist. Aus diesem Signal wird durch einen kurzen positiven Impuls auf der Leitung 49 ein Impuls ausgeblendet, der in Fig. 3a mit einer ausgezogenen Linie dargestellt ist und der u. a. auch dafür sorgt, daß bei bestimmten Bedingungen Störimpulse zuverlässig vermieden werden. Aus Fig. 3 ist zu erkennen, daß der OFL-Impuls auch erzeugt wird, wenn der Programmzähler 8 in Fig. 1 durch einen Sprung von einem hohen Wert auf einen niedrigen Wert geht, solange dieser Sprung nicht zu groß wird, d. h. solange sich beide Adreßbits A14 und A15 von einem hohen zu einem niedrigen Wert ändern.

Ein entsprechender Vorgang erfolgt auch beim Rückwärtszählen bzw. hier bei einem Rückwärtssprung des Programmzählers 8 von einer niedrigen Stellung am Anfang eines Segments in eine hohe Stellung am Ende eines vorhergehenden Segments. In der niedrigen Stellung des Programmzählers 8 haben beide Adreßbit A14 und A15 einen niedrigen Wert. Bei einem begrenzten Rückwärtssprung über die Segmentgrenze nehmen beide Adreßbit A14 und A15 einen hohen Wert an, während der Ausgang QN des Flipflops 40 noch einen hohen Wert für eine Taktphase behält. Damit tritt an dem mit UFL (Borgen) bezeichneten Ausgang 11b ein Signal auf, das in Fig. 3b gestrichelt dargestellt ist. Auch hier wird durch einen kurzen positiven Impuls auf der Leitung 49 ein Impuls ausgeblendet, der in Fig. 3b mit einer ausgezogenen Linie dargestellt ist.

Aus der in Fig. 2 dargestellten Schaltung ist leicht zu erkennen, daß durch ein hohes Signal auf der Leitung 23 jedes Ausgangssignal OFL bzw. UFL unterdrückt werden kann.

Das muß für jeden Sprungbefehl erfolgen, mit dem innerhalb des gesamten Segment-Adreßbereichs gesprungen werden kann, also beispielsweise vom Segment-Anfang zum -Ende. Diese Befehle sind in Fig. 1 auf den Leitungen 21 dekodiert und im ODER-Gatter 22 verknüpft und werden über die Leitung 23 der Zähltakt-Schaltung 10 zugeführt.

Die in Fig. 2 dargestellte Schaltung kann im übrigen auch für beliebige Zähler, insbesondere bei sehr langen Zählern, verwendet werden, um ein Übertragssignal oder ein Borgsignal zu erzeugen, ohne die gesamte Zählerstellung dekodieren zu müssen.

Patentansprüche

1. Mikroprozessor mit einer Verarbeitungseinheit (20) zum Empfangen und Entschlüsseln von Befehlen, einem Befehlsspeicher (30) zur Abgabe von Befehlen und einem Programmzähler (8) vorgegebener Kapazität zum Adressieren des Befehlsspeichers, **dadurch gekennzeichnet,** daß zur Erweiterung der maximalen Größe des Befehlsspeichers ein erstes Register (2), von dem parallele Ausgänge mit einem Adreßausgang (7b) des Mikroprozessors (1) für zusätzliche höchstwertige Adreßbits des Befehlsspeichers (30) gekoppelt sind, sowie eine Zähltakt-Schaltung (10) vorgesehen sind und die Zähltakt-Schaltung an Adreßausgänge des Programmzählers (10) angeschlossen ist und einen Aufwärts-Zähltakt für das erste Register (2) erzeugt, wenn der Programmzähler (8) seine höchste Stellung überschreitet, und einen Abwärts-Zähltakt für das erste Register erzeugt, wenn der Programmzähler seine niedrigste Stellung unterschreitet, und daß ein zweites Register (4) vorgesehen ist, von dem parallele Eingänge an einen internen Bus (12) angeschlossen sind zum Übernehmen des auf dem internen Bus vorhandenen Werts bei einem vorgegebenen Schreibbefehl und von dem parallele Ausgänge mit parallelen Eingängen des ersten Registers (2) verbunden sind zur Übertragung des Inhalts des zweiten Registers in das erste Register bei Adreßsprungbefehlen.

2. Mikroprozessor nach Anspruch 1, dadurch gekennzeichnet, daß die parallelen Ausgänge des ersten Registers (2) über UND-Gatter (6) mit dem Adreßausgang (7b) verbunden sind und ein gemeinsamer Steuereingang (6a) der UND-Gatter (6) mit einer Interrupt-Schaltung (28, 33, 34) verbunden ist, die die UND-Gatter wenigstens zu Beginn der Verarbeitung eines Interrupt-Signals sperrt.

3. Mikroprozessor nach Anspruch 2, dadurch gekennzeichnet, daß die Interrupt-Schaltung (28, 32) ein erstes Speicherglied (28) enthält, das beim Auftreten eines Interrupt-Signals gesetzt und bei der Durchführung eines Adressensprungs zurückgesetzt wird und dessen Ausgang mit dem Steuereingang (6a) der UND-Gatter (6) über ein NAND-Gatter (32) gekoppelt ist, von dem ein weiterer Eingang mit der Verarbeitungseinheit (20) zum Empfangen eines Signals während der Ausführung einer Interrupt-Routine gekoppelt ist.

4. Mikroprozessor nach einem der Ansprüche 1 bis 3, dadurch gekennzeichnet, daß der Ausgang des ersten Registers (2) mit dem internen Bus (12) koppelbar ist.

5. Mikroprozessor nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die Zähltakt-Schaltung (10) mit Ausgängen für die beiden höchstwertigen Adreßbits (A15, A14) des Programmzählers (8) gekoppelt ist und eine zweite Speicherstufe (40), die den Signalzustand des höchstwertigen Adreßbits (A15) für eine vorgegebene Zeitdauer nach einem Zustandswechsel dieses Adreßbits speichert, sowie zwei Gatter (42, 44) mit je wenigstens drei Eingängen enthält, wobei die Eingänge jedes Gatters mit wenigstens einem Ausgang (41) der Speicherstufe (40) sowie mit den Ausgängen für die beiden höchstwertigen Adreßbits (A15, A14) derart gekoppelt sind, daß das eine Gatter (42) einen Aufwärts-Zähltakt abgibt, wenn beide Adreßbits ihren Zustand von hoch nach niedrig wechseln, und das andere Gatter (44) einen Abwärts-Zähltakt abgibt, wenn beide Adreßbits ihren Zustand von niedrig nach hoch wechseln.

Hierzu 2 Seite(n) Zeichnungen

Nummer: DE 43 40 551 A1
Int. Cl.[6]: G 06 F 12/06
Offenlegungstag: 1. Juni 1995

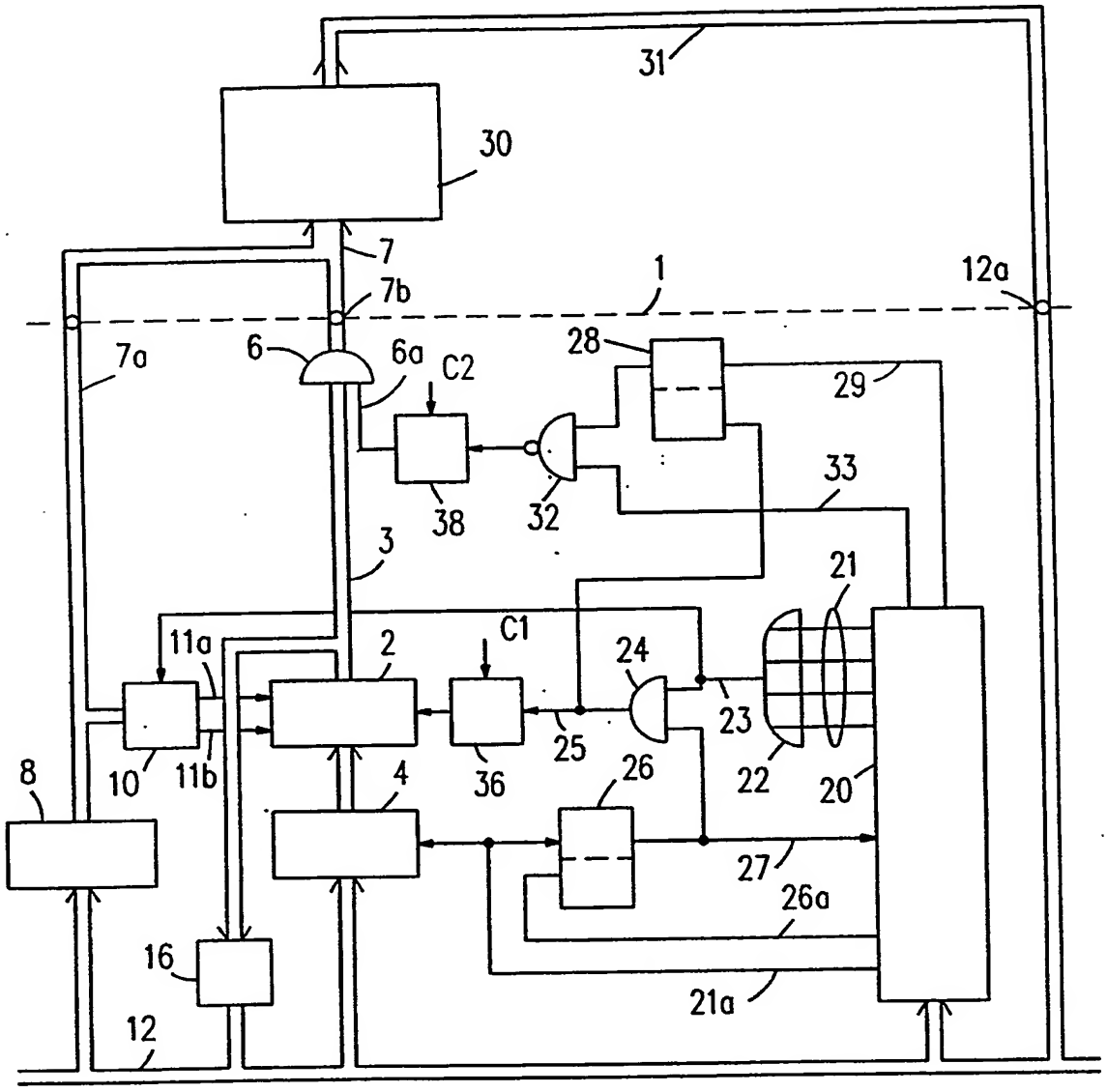


Fig.1

Nummer: DE 43 40 551 A1
Int. Cl.[6]: G 06 F 12/06
Offenlegungstag: 1. Juni 1995

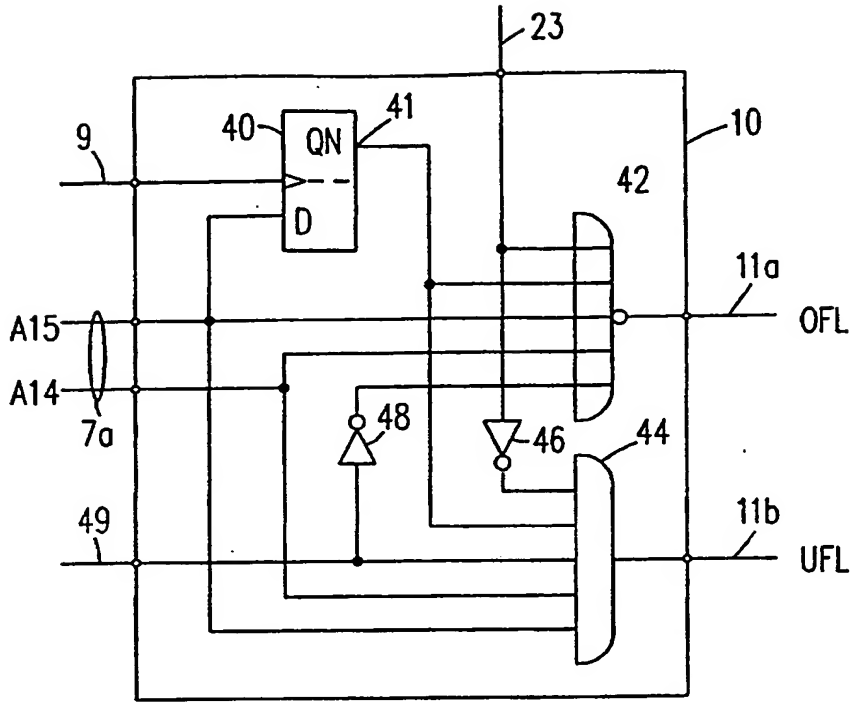
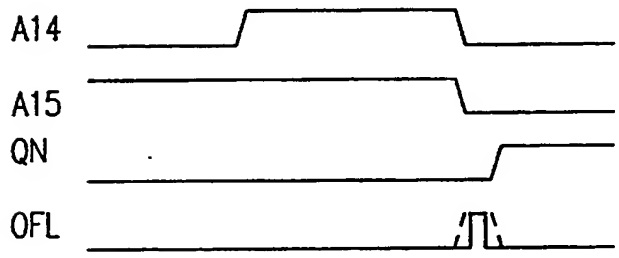
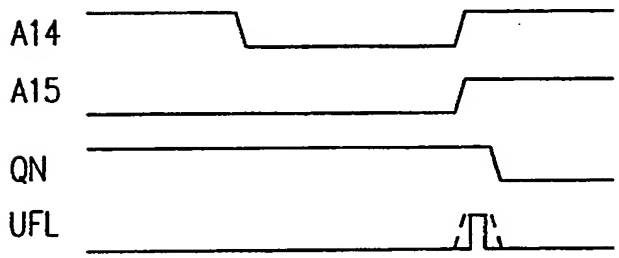


Fig.2



a



b

Fig.3